REMARKS:

Claim 1 was presented for examination and was pending in this application. In an Official Action dated June 1, 2004, claim 1 was rejected. Applicants thank Examiner for examination of the claim pending in this application and addresses Examiner's comments below.

Applicants herein add new claims 2-27. These changes are believed not to introduce new matter, and their entry is respectfully requested. Based on the following Remarks, Applicants respectfully request that Examiner reconsider all outstanding objections and rejections, and withdraw them.

Objection to the Oath/Declaration

The Examiner has objected to the Oath/Declaration as being defective for lack of identification of city and either state or foreign country of residence of each inventor. As suggested by Examiner, Applicants submit herewith an application data sheet with the required information. Accordingly, Applicants respectfully request that Examiner withdraw the objection to the Oath/Declaration.

Objection to the Title

The Examiner has objected to the title as not being descriptive. Applicants have amended the title and now believe the title to be clearly indicative of the invention to which the claims are directed. In particular, Applicants have amended the title to indicate that the fixed length memory-to-memory arithmetic and architecture of the present invention can be implemented in a communications embedded processor system. These are features that distinguish the present invention from conventional general-purpose microprocessor schemes. Accordingly, Applicants respectfully request that Examiner reconsider and withdraw the objection to the title.

The Drawings

Applicants thank Examiner for kindly reviewing and accepting the formal drawings filed in this application on June 22, 2001.

Response to Rejection Under 35 USC 103(a) in View of Intel and Case

In the 5th paragraph of the Office Action, Examiner rejects claim 1 under 35 USC § 103(a) as allegedly being unpatentable in view of Intel's Pentium Processor Family Developer's Manual, Vol. 3: Architecture and Programming Manual, 1995, ("Intel") and U.S. Patent No. 4,777,587 to Case et al. ("Case"). This rejection is respectfully traversed.

Based on the following Remarks, Applicants respectfully submit that for at least these reasons claim 1 is patentably distinguishable over the cited references, both alone and in combination. Therefore, Applicants respectfully request that Examiner reconsider the rejection, and withdraw it.

Claim 1 recites in part:

> A method for performing arithmetic in a memory to memory architecture in an embedded processor, the method comprising:
>
> receiving a 32-bit fixed length instruction, the instruction specifying a source address in a memory using 11 bits, a source address in a register file using 5 bits, <u>a destination address in the memory</u> using 11 bits, and a mathematical operation to be performed using 5 bits.

(Emphasis added). Claim 1 recites receiving a fixed length instruction that specifies a source address and a destination address in memory. A fixed-length instruction set allows for fast pipeline processing. At the same time, the ability to access data from memory directly without having to load it to a register is important. The use of fixed-length instructions capable of direct memory access enables very high performance when processing network traffic.

The Intel reference describes the instruction set for the Pentium® processor family. As Examiner points out, the Intel reference does not show a fixed length instruction; Intel's Pentium architecture is based on a variable length instruction set. Further, the "ADD" instruction referenced to by Examiner also fails to show or suggest (1) "a source address in a memory," (2) "a source address in a register file," and (3) "a destination address in the memory." In particular, the reference does not show both a source address and a destination address in memory that are not the same. The Intel reference describes that "[t]he ADD instruction performs an integer addition of the two operands (DEST and SRC). The result of the addition <u>is assigned to the first operand</u> (DEST), and the flags are set accordingly." Intel, p.25-29 (emphasis added). Since the result is assigned to the first operand, there is no need to specify an additional memory address, as the claim requires. Further, the Intel reference describes that its two-operand instructions, such as the ADD instruction, are not memory to memory and that memory to memory data transfers cannot be accomplished directly as claim 1 recites but rather indirectly either implicitly through a memory-based stack or using the registers:

> For most instructions, one of the two explicitly specified operands—either the source or the destination—can be either in a register or in memory. The other operand must be in a register or it must be an immediate source operand. This puts the explicit two-operand instructions into the following groups:
>
> · Register to register
>
> · Register to memory
>
> · Memory to register

> · Immediate to register
>
> · Immediate to memory
>
> Certain string instructions and stack manipulation instructions, however, transfer data from memory to memory. Both operands of some string instructions are in memory and are specified implicitly. Push and pop stack operations allow transfer between memory operands and the memory-based stack.
>
> Several three-operand instructions are provided, such as the IMUL, SHRD, and SHLD instructions. Two of the three operands are specified explicitly, as for the two-operand instructions, while a third is taken from the CL register or supplied as an immediate. Other three-operand instructions, such as the string instructions when used with a repeat prefix, take all their operands from registers.

Intel at p. 3-18. (Although Applicants presume that the entire Intel reference is in front of the Examiner, for Examiner's convenience a copy of this page is attached herewith).

The Case reference describes an instruction processor suitable for use in a reduced instruction-set computer. See Case at Abstract. The instructions discussed in the Case reference are for RISC type processor architectures as opposed to the complex instruction set computing ("CISC") architecture used by Intel® processors. Further, the instructions discussed in the Case reference are not capable of direct memory-to-memory arithmetic as recited in the claimed invention. Reduced instruction sets used in RISC architectures are based on register-to-register operations typically requiring additional "load" and "save" memory instructions. The Case reference fails to show or suggest the use of (1) "a source address in a memory," (2) "a source address in a register file," and (3) "a destination address in the memory" as recited in claim 1.

Therefore, the Examiner has failed to point out any prior art teaching which anticipates or renders obvious the explicit recitation in the language of claim 1 of "receiving a 32-bit fixed length instruction, the instruction specifying a source address in a memory using 11 bits, a source address in a register file using 5 bits, a destination address in the memory using 11 bits, and a mathematical operation to be performed using 5 bits." Therefore, it is respectfully submitted that the rejection is improper and should be withdrawn.

In addition, 35 U.S.C. § 103 authorizes a rejection where, to meet the claim, it is necessary to modify a single reference or to combine it with one or more others. After indicating that the rejection is under 35 U.S.C. §103, there should be set forth [by Examiner] "(1) the difference or differences in the claim over the applied references, (2) the proposed modification of the applied references necessary to arrive at the claimed subject matter, and (3) an explanation of <u>why</u> such proposed modification would be obvious." MPEP § 706.02; emphasis added.

The Examiner states that it would have been obvious at the time of the invention was made to have the instructions of Intel be of fixed-length, as taught by Case, for the desirable purpose of simplifying instruction decoding. However, the Intel variable instruction set is based on the fact that the instructions are variable and uses this potential variation to provide enhanced and more complex functionality supported by the higher complexity architecture of the Pentium® processors. As Case describes, most techniques achievable by these more advanced processors are too complex for a RISC architecture. See Case, col. 1, lines 29-30. The Examiner does not provide any information as to how the Intel instruction set could possibly be modified as the Examiner suggests. Neither does the Examiner provide a explanation as to why such proposed modification would be obvious; whether simplifying instruction decoding is desirable does not provide a an explanation as to why modifying the highly complex Intel instruction set to be made up of fixed-length instructions would be obvious. For example, the ability to specify the width of operands as well as the width of the arithmetic by the instruction set itself is based on the variable nature of the Intel instruction set. If, as the Examiner suggests, the variable length instruction set of Intel is modified to be a fixed-length instructions set as in Case, the Intel instructions relying on the ability to vary their length could not properly function.

In support of the combination, Examiner contends that "a person in the art would be motivated to combine the cited references because of the desirable purpose of simplifying instruction decoding." In order to support a rejection under 35 USC § 103, however, the Examiner must provide "some objective teaching in the prior art or that knowledge generally available to one of ordinary skill in the art would lead that individual to combine the relevant teachings of the references." In re Fine. The Examiner has not cited an objective prior art reference that provides an incentive, motivation, or suggestion for making the suggested combination. Also, Examiner has not established by objective evidence that knowledge generally available to one of ordinary skill in the art would lead one to make the suggested combination. Thus, Applicants respectfully assert that the suggested combination is improper.

It is well settled law that when making a rejection under 35 U.S.C. § 103, Examiner has the burden of establishing a prima facie case of obviousness. Examiner can satisfy this burden "only by showing some objective teaching in the prior art or that knowledge generally available to one of ordinary skill in the art would lead that individual to combine the relevant teachings of the references" in the manner suggested by Examiner. In re Fine, 5 USPQ2d 1596, 1598 (Fed. Cir. 1988). "[E]lements of separate prior patents [and/or publications] cannot be combined when there is no suggestion of such combination anywhere in those patents [and/or publications]...; and a court should avoid hindsight..." (emphasis added; annotations within square brackets). Panduit Corp. v. Dennison Mfg. Co., 1 USPQ2d 1593, 1597 (Fed. Cir. 1987), citing ACS Hospital Systems, Inc. v. Montefiore Hospital, 220 USPQ 929, 933 (Fed. Cir. 1984), and W.L. Gore & Associates v. Garlock, Inc., 220 USPQ 303, 313 (Fed. Cir. 1983). See also Uniroyal Inc. v. Rudkin-Wiley Corp., 5 USPQ2d 1434, 1438-1441 (Fed. Cir. 1988). In fact, it is impermissible to use hindsight reconstruction to pick and choose among isolated disclosures in the prior art to deprecate the claimed invention. In re Fine, 5 USPQ2d at 1600.

For all of the above reasons, Applicants respectfully assert that claim 1 is patentable over Intel and/or Case, and that the combination is improper in any case, and therefore respectfully request that Examiner reconsider and withdraw the rejection.

Conclusion

Applicants have added new claims 2-17 for which Applicants request consideration and examination. Applicants respectfully submit that these are supported by the specification and are commensurate within the scope of protection to which Applicants believe they are entitled.

In sum, Applicants respectfully submit that claims 1 through 27, as presented herein, are patentably distinguishable over the cited references. Therefore, Applicants request reconsideration of the basis for the rejection to claim 1 and request allowance of all of the claims.

In addition, Applicants respectfully invite Examiner to contact Applicants' representative at the number provided below if Examiner believes it will help expedite furtherance of this application.

Respectfully Submitted,
DAVID A. FOTLAND, ET AL.

Date: Aug. 31, 2004 By: HRibera

Hector J. Ribera, Attorney of Record
Registration No. 54,397
FENWICK & WEST LLP
801 California Street
Mountain View, CA 94041
Phone: (650) 335-7192
Fax: (650) 938-5200
E-Mail: hribera@fenwick.com

result. This is one difference between the *source operand* (the one unaffected by the operation) and the *destination operand* (the one overwritten by the result).

For most instructions, one of the two explicitly specified operands—either the source or the destination—can be either in a register or in memory. The other operand must be in a register or it must be an immediate source operand. This puts the explicit two-operand instructions into the following groups:

- Register to register
- Register to memory
- Memory to register
- Immediate to register
- Immediate to memory

Certain string instructions and stack manipulation instructions, however, transfer data from memory to memory. Both operands of some string instructions are in memory and are specified implicitly. Push and pop stack operations allow transfer between memory operands and the memory-based stack.

Several three-operand instructions are provided, such as the IMUL, SHRD, and SHLD instructions. Two of the three operands are specified explicitly, as for the two-operand instructions, while a third is taken from the CL register or supplied as an immediate. Other three-operand instructions, such as the string instructions when used with a repeat prefix, take all their operands from registers.

3.5.1. Immediate Operands

Certain instructions use data from the instruction itself as one (and sometimes two) of the operands. Such an operand is called an *immediate* operand. It can be a byte, word, or doubleword. For example:

```
SHR PATTERN, 2
```

One byte of the instruction holds the value 2, the number of bits by which to shift the variable PATTERN.

```
TEST PATTERN, 0FFFF00FFH
```

A doubleword of the instruction holds the mask which is used to test the variable PATTERN.

```
IMUL CX, MEMWORD, 3
```

A word in memory is multiplied by an immediate 3 and stored into the CX register.

All arithmetic instructions (except divide) allow the source operand to be an immediate value. When the destination is the EAX or AL register, the instruction encoding is one byte shorter than with the other general registers.